UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/802,309	03/17/2004	Alex Brown	1801270.00139US1	3911

23483 7590 12/17/2007

WILMERHALE/BOSTON
60 STATE STREET
BOSTON, MA 02109

EXAMINER	
VO, TED T	

ART UNIT	PAPER NUMBER
2191	

NOTIFICATION DATE	DELIVERY MODE
12/17/2007	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

michael.mathewson@wilmerhale.com
teresa.carvalho@wilmerhale.com
sharon.matthews@wilmerhale.com

Office Action Summary	Application No.	Applicant(s)	
	10/802,309	BROWN ET AL.	
	Examiner	Art Unit	
	Ted T. Vo	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *10 September 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-63* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-63* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-63 remain pending in the application.

Response to Arguments

2. This is in response to the arguments filed on 09/10/2007.

With regards to the argument to the rejection of claims 1, 22, 43 as being anticipated by Muth: Applicants argued Muth does not describe grouping the subject code, the pages 140 and 40 do not relate to grouping of subject code to account for self-modifying code.

Examiner's response: The argument fails to explain the method to differ in term of patentability from the existing method in the prior art while the claims remain reciting all common techniques used in compilation. Particularly, it is the common technique that modifies the code for register allocation. Argument in the preamble, "*...to account for self-modifying code*", does not make a distinction because it is only an intended use. It should be noted that code grouping is common in compiling, it is not new. The purpose for it is to meet with the limited number of registers in a target processor. In P. 140, it addresses: "These sets of basic block are then partitioned (i.e., *Accounting for self-modifying code*) into groups of identical regions". In P. 40, and thereafter, it addresses a self Modifying code as a piece of code A changed into B. Muth addressed all the common techniques in the compilation related to Self Modifying Code. The common act of parsing such as Figure 2.1, p. 31, is identifying the self-modifying code and it is shown similarly in the recitation of the claim. Dividing memory region as in the claims reads on Address Translation in p. 41. In the remarks, the reasonable argument in term of patentability is absent, as it is required by 1.111(c).

With regards to the argument to the rejection of claims 1-63 as being anticipated by Hsu:

Applicants argued Hsu does not teach or suggest a method of grouping subject code during a translation of subject code into translate target code to account for self modifying subject code.

Examiner's response: As noted above, grouping of code in compilation is not new. In compilation, particularly in self modifying code, subject code will be translated into a target code (i.e. Code A becomes code B). Hsu (or Muth) is addressing "Self-Modifying Code"; the translations in Hsu is accounting for self-modifying subject code into a native instructions of a processor. See p. 7 (second full paragraph): "*This table maps groups of base instructions to groups of native instructions..*". The groups of base instructions are the subject code grouped for translation. It refers seeing throughout the reference as well, where the reference teaches new code that is modified and mapped into addresses of a memory. For example, p. 30 (Fig. 4.5), "allocated a memory space Mem[i]", and this memory allocation is part of Self-Modifying Code as discussed in the Chapter 6; it meets the generic limitation "diving a region of memory" in the claims.

Because Applicants fail to address the patentable difference in the claims, but the arguments are merely addressing the common acts used in the compilation, the arguments fail under 1.111(c).

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1, 22, 43 are rejected under 35 U.S.C. 102(b) as being anticipated by Muth, "ALTO: A Platform for Object Code Modification", 1999.

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per Claim 1: Muth discloses,

A method of grouping subject code (e.g., basic blocks, p. 140) *during a translation of subject code into translated target code to account for self-modifying subject code* (See p. 40, sec. 2.3, i.e. "Code Generation"), *comprising:*

identifying self-modifying code events in said subject code during translation of subject code into translated code and also during subsequent execution of translated code (See p. 40, the discussion of parsing a piece code A in a Text Segment of Figure 2.1 (p.31). Also see discussion Address Translation start in p. 41, and further see p. 42, refer to memory location containing code addresses, and Figure 2.4 represent a Text Segment that has self-modifying code events such as targetA, targetB, targetC)); *and*

dividing a region of memory containing said subject code into at least one subject instruction group of subject addresses *when* ***identifying a self-modifying code event, wherein each subject instruction group includes one or more ranges of subject code addresses in said memory which are affected by a respective self-modifying code event*** (See discussion Address Translation started in p. 41, and see the modifying the basis blocks, in p. 135).

As per Claim 22: See rationale addressed in the rejection of Claim 1.

As per Claim 43: See rationale addressed in the rejection of Claim 1.

5. Claims 1-63 are rejected under 35 U.S.C. 102(b) as being anticipated by Hsu, "A Robust Foundation for Binary Translation of X86 Code", 1997.

As per Claim 1: Hsu discloses,

A method of grouping subject code during a translation of subject code into translated target code to account for self-modifying subject code comprising:

identifying self-modifying code events in said subject code during translation of subject code into translated code and also during subsequent execution of translated code (See p. 3, "self-modifying code detector"); ***and***

dividing a region of memory (See p. 26: Control flow analysis: Control flow analysis is an element used in code translation; its task is to partition subject code into basic blocks, and latter involved in memory allocation. Also see p. 30, sec. 4.4, "allocated memory") ***containing said subject code into at least one subject instruction group of subject addresses*** *when* ***identifying a self-modifying code event, wherein each subject instruction group includes one or more ranges of subject code addresses in said memory which are affected by a respective self-modifying code event*** (See p. 26: lines 1-4, the translator merges them to form a larger single area).

As per Claim 2: Hsu discloses, *The method of claim 1, wherein each subject instruction group is further associated with translated target code corresponding to subject code contained in that subject instruction group* (i.e. "code generator" discussed in p. 2).

As per Claim 3: Hsu discloses, *The method of claim 1, wherein each said subject instruction group represents a region of memory that does not overlap with regions of memory described by other subject instruction groups* (i.e. "allocated memory " discussed in p. 30).

As per Claim 4: Hsu discloses, *The method of claim 1, wherein each said subject instruction group represents a region of memory that may overlap with regions of memory contained in other subject instruction groups* (i.e. a merged region into a "larger single area" discussed in p. 26).

As per Claim 5: Hsu discloses, *The method of claim 1, wherein a self-modifying code event modifies a respective range of subject code addresses, said method further comprising: modifying subject instruction*

groups existing in said memory that contain subject code addresses which are affected by said self-modifying code event (see chapter 6, the discussion of code that is modified. In p. 57-58, discussing linear address that may exist self-modifying code (p. 58: last paragraph)).

As per Claim 6: Hsu discloses, *The method of claim 5, wherein said subject instruction group modifying step comprises: creating a new subject instruction group to include subject code addresses containing modified subject code corresponding to the self-modifying code event* (Start at p. 66, see chapter 7, Post Translation, and Figure 7.1. Also See p. 30:1-3: code segment will be updated when translator creates the new executable file); *and*

for existing subject instruction groups having ranges of subject code addresses which overlap with the subject code addresses of the newly created subject instruction group, modifying said existing subject instruction groups to delete the subject code addresses from said existing subject instruction groups that overlap with the subject code addresses of the newly created subject instruction groups such that the subject instruction groups no longer overlap (See associated discussion of Figure 7.1 (p.70) and also refer to the merged region discussed in p. 26).

As per Claim 7: Hsu discloses, *The method of claim 6, wherein each subject instruction group is further associated with translated target code corresponding to subject code contained in that subject instruction group, said method further comprising: deleting translated target code associated with subject instruction groups that have been modified in response to the self-modifying code event; and translating new target code for the subject code contained in the modified subject instruction groups* (See Chapter 7, Post Translation, start at p. 66, and particularly Figure 7.1, disclose creating a new code segment (New Segment 1) from old subject instruction group, and the translation of New segment).

As per Claim 8: Hsu discloses, *The method of claim 6, further comprising associating translated target code with a subject instruction group as its corresponding subject code contained in that subject instruction group is translated* (See Figure 7.1).

As per Claim 9: Hsu discloses, *The method of claim 8, wherein each subject instruction group includes a particular range or ranges of subject code addresses that have been translated, such that the particular ranges of subject code addresses having been translated comprises an active sub-group within the*

subject instruction group, said method further comprising: determining whether the subject code addresses of said newly created subject instruction group overlap with any subject code addresses in said active sub-group of any existing subject instruction group; and for existing subject instruction groups having an active sub-group that overlaps with the subject code addresses of said newly created subject instruction group, deleting translated target code associated with subject instruction groups that have been modified in response to the self-modifying code event, and translating new target code for the subject code contained in the modified subject instruction groups (Refer Chapter 7, and see Figure 7.1).

As per Claim 10: Hsu discloses, *The method of claim 9, wherein each subject instruction group includes a range or ranges of subject code addresses that have not been translated referred to as an inactive sub-group within the subject instruction group, said method further comprising: for existing subject instruction groups having an active sub-group which does not overlap with the subject code addresses of said newly created group but having an inactive sub-group that does overlap with the subject code addresses of said newly created subject instruction group, modifying said existing subject instruction groups to delete the subject code addresses from said inactive sub-groups in said existing subject instruction groups that overlap with the subject code addresses of the newly created subject instruction group such that the subject instruction groups no longer overlap, and leaving the translated target code associated with active sub-groups in said existing groups unchanged* (Refer Chapter 7, and see Figure 7.1).

As per Claim 11: Hsu discloses, *The method of claim 5, further comprising: identifying subject instruction groups that are adjacent to one another in memory having characteristics that allow them to be combined; and aggregating said adjacent subject instruction groups into a single, combined subject instruction group* (See p. 26:1-4).

As per Claim 12: Hsu discloses, *The method of claim 1, wherein said self-modifying code event is identified during decoding of the subject code, said method further comprising inserting a special translation structure into a control flow of the translated target code as a representation of the identified self-modifying code event* (start at p. 25: See sec. 4.2.2).

As per Claim 13: Hsu discloses, *The method of claim 12, in response to encountering said special translation structure during execution of the translated target code, said method further comprising:*

identifying the range or ranges of subject code addresses affected by the self-modifying code event, and creating the subject instruction group in memory using this identified range of subject code addresses (p. 26:1-4; see adjacent instruction areas. See Chapter 7, start at p. 66, discloses creating the modification of self-modifying code within the modifying region, for example, see Figure 7.1).

As per Claim 14: Hsu discloses, *The method of claim 1, further comprising identifying control flow instructions in the current subject instruction group which represent an actual or possible transfer of control to subject addresses outside the current subject instruction group* (See discussion of Control-flow Analysis start at p. 25).

As per Claim 15: Hsu discloses, *The method of claim 14, wherein said control flow instruction is identified during decoding of the subject code, said method further comprising inserting a special exit translation structure into the control flow of the translated target code as a representation of the identified control flow event* (See discussion of Control-flow Analysis start at p. 25).

As per Claim 16: Hsu discloses, *The method of claim 15, wherein control flow that passes from subject code in one subject instruction group into subject code in a different, second subject instruction group is represented using a pair of special translation structures, wherein said pair of special translation structures includes said exit structure and also an entry structure, such that each exit structure contains a specific reference to a counterpart entry structure associated with succeeding subject instruction group to be executed next* (See discussion of Control-flow Analysis start at p. 25, and refer to p. 34, Figures 4.9-10).

As per Claim 17: Hsu discloses, *The method of claim 16, when encountering an exit structure during execution of target code associated with a current subject instruction group, said method further comprising verifying that a counterpart entry structure exists in a successive subject instruction group before passing control from the current partition to the successive group* (See discussion of Control-flow Analysis start at p. 25, and refer to p. 34, Figures 4.9-10).

As per Claim 18: Hsu discloses, *The method of claim 17, when encountering an exit structure during execution of target code associated with a current subject instruction group, wherein said exit structure is not associated with a counterpart entry structure existing in a successive subject instruction group,*

creating such an entry structure and associating it with the appropriate successive subject instruction group which contains the successive subject address to be executed, and modifying said exit structure to specifically refer to said newly created entry structure (See discussion of Control-flow Analysis start at p. 25, and refer to p. 34, Figures 4.9-10).

As per Claim 19: Hsu discloses, *The method of claim 16, wherein a set of border guards exists containing exit structures and entry structures for all partitions, said method further comprising modifying said set of exit structures and entry structures whenever a subject instruction group is deleted in response to a self-modifying code event* (See discussion of Control-flow Analysis start at p. 25, and refer to p. 34, Figures 4.9-10).

As per Claim 20: Hsu discloses, *The method of claim 5, wherein when subject code defines a multi-threaded program, said method further comprising preventing other threads from entering a subject instruction group while the subject instruction group is being modified by another thread* (Refer to instruction set x86, multi-tasking (p. 78), and the discussion of self-modifying code).

As per Claim 21: Hsu discloses, *The method of claim 5, wherein each subject instruction group is further associated with translated target code corresponding to subject addresses contained in that subject instruction group, wherein each partition includes a set of entry structures and exit structures represent control flow passing between subject instruction groups, such that each exit structure contains a specific reference to a counterpart entry structure in a succeeding subject instruction group to be executed next, said method further comprising: providing a memory management subsystem having regions which mirror the subject instruction groups, wherein said memory management subsystem stores target code and entry structures and exit structures associated with a subject instruction group along with its corresponding target code; and deleting an entire region of said memory management subsystem that corresponds to a specific subject instruction group whenever that specific subject instruction group is modified* (See Figure 4.9-10, it show an example of partition segments that comprises entry/exit structure to such segments ('ofs'). The modification of segments, for example, segment 2, will cause the content being deleted under trap, and the management of trap will cause corresponding to the modified

instructions at an available region within the memory. Such modifying is consistent to the discussion within chapter 7).

As per Claims 22-42: See rationale addressed in the rejection of Claims 1-21, respectively.

As per Claims 43-63: See rationale addressed in the rejection of Claims 1-21, respectively.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 4:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV
December 07, 2007

TED VO
PRIMARY EXAMINER